CLAIMS:

1. A data processor for executing, instructions realized by wired logic, in a pipeline system, comprising:

a program counter for indicating an address of an instruction to be read;

m(m = 2, 3, ...) instruction registers for storing a sequency of m instructions of which the first address is pointed by said program counter, at a time in one machine cycle;

a register file formed of a plurality of registers in which data are stored; and

m arithmetic operation units which read corresponding data from said register file and make arithmetic operations thereof in parallel in accordance of instructions read from said instruction registers;

whereby the contents of said program counter is increased by m at a time, sequential m instructions are read from the addresses pointed by said program counter and written in said instruction registers, and said m instructions are processed in parallel by said m arithmetic operation units.

2. A data processor according to claim 1, wherein said instructions have a fixed length, and the number of instructions to be processed in one machine cycle in satisfies $m = 2^n$ (n = 1, 2, ...).

3. A data processor according to claim 1, further comprising:

means for detecting whether one instruction of said m instructions has a conflict with a subsequent instruction or not; and

means for controlling to substantially execute only said one instruction in one machine cycle when said detection means detected said conflict.

4. A data processor according to claim 3, wherein while only aid one instruction is substantially being executed in one machine cycle, said control means controls the other instructions to be executed with NOP instruction.

5. A data processor according to claim 3, wherein said detection means detects said conflict by deciding whether or not said subsequent instruction reads the register in which data is written by said one instruction.

6. A data processor according to claim 3, wherein said detection means is constructed to access only once to a memory in one machine cycle and detects said conflict by deciding whether each of said one instruction and said subsequent instruction is load instruction or store instruction.

7. A data processor according to claim 3 wherein said detection means detects said conflict on the basis of the facts that some arithmetic operation units of said m arithmetic operation units have a barrel shifter, that at least one of said one instruction and said subsequent instruction is a bit shift instruction, and that the arithmetic operation unit for processing said bit shift instruction has no barrel shifter.

8. A data processor according to claim 3, further comprising:

a cache memory for instructions which is provided before said instruction registers;

wherein said detection means includes, means for writing a conflict bit indicating said conflict at every m instructions which are read at a time, within said cache memory.

9. A data processor according to claim 3, further comprising:

a cache memory for instructions which is provided before said instruction registers;

wherein said detection means includes means for monitoring at each machine cycle the conflict between instructions produced from said cache memory.

10. A data processor according to claim 1, wherein an instruction for altering a status flag and a branch

instruction of which the branch condition is satisfied by said status flag is included in the m instructions read by said program counter, and there is provided means for detecting presence of a conditional branch instruction at a predetermined address to suppress execution of instructions after a corresponding address of the m instructions when the condition is satisfied.

11. A data processor according to claim 1, wherein part of the memory in which instructions are stored is formed of an ROM.

12. A data processor according to claim 1, wherein any interrupt is inhibited between the m instructions which are read at a time.

13. A data processor for executing, an instruction realized by wired logic, in a pipeline system, comprising: a program counter for indicating the address of instructions to be read;

m(m = 2, 3, ...) instruction registers for storing in one machine cycle at a time sequential m instructions of which the first address is indicated by said program counter;

a register file formed of a plurality of registers in which data is stored; and

m arithmetic operation units for reading corresponding data from said register file and making

arithmetic operation in parallel in accordance with the instructions read from said instruction registers;

whereby only when the contents of said program counter is within a limited range, sequential m instructions of which the first address is indicated by said program counter are read and written in said instruction registers, said m instructions are processed by said m arithmetic operation units, and when the contents of said program counter is out of said range, ℓ ($\ell > m$, $\ell = 1, 2, \ldots$) instructions are read in said instruction registers and processed in parallel by Z arithmetic operation units.

14. A data processor according to claim 13, wherein said ℓ satisfies $\ell = 1$.

15. A data processor according to claim 13, wherein said data processor is all integrated on a one-chip semiconductor substrate, and at least part of a program within said range is integrated in a form of ROM on said semiconductor substrate.

16. A data processor for executing, an instruction realized by wired logic, in a pipeline system, comprising:

a program counter for indicating an address of instructions to be read;

m(m = 2,3, ...) instruction registers for storing in

one machine cycle at a time sequential m instructions of which the first address is indicated by said program counter;

a register file formed of a plurality of registers in which data are stored; and

m arithmetic operation units for reading corresponding data from said register file and making arithmetic operation of said data in parallel in accordance with said instructions read from said instruction registers in one machine cycle at a time.